PLANNING AND RESOURCE MANAGEMENT IN AN INTELLIGENT AUTOMATED POWER MANAGEMENT SYSTEM

Final Report

NASA/ASEE Summer Faculty Fellowship Program -- 1991

Johnson Space Center

Prepared By:	Robert A. Morris, Ph.D.
Academic Rank:	Assistant Professor
University and Department:	Florida Institute of Technology Computer Science Dept. Melbourne, Florida 32901
NASA/JSC Directorate:	Engineering
Division:	Propulsion and Power
Branch:	Power
JSC Colleague:	Thomas D. Jeffcoat
Date Submitted:	August 23, 1991
Contract Number:	NGT-44-001-800

ABSTRACT

Power system management is a process of guiding a power system towards the objective of the continuous supply of electrical power to a set of loads. Spacecraft power system management, unlike most ground-based procedures, requires planning and scheduling, since electrical power is typically a scarce resource in space. This fact will be most evident in future longer-term space missions such as Space Station Freedom, lunar and Mars bases. The automation of power system management for future spacecraft has been recognized as an important research and development goal. Current automation technology has evolved to include microprocessing capabilities in switchgear, applying data and numerical processing capabilities to the retrieval, storage, and analysis of sensor and switchgear data, and the use of expert systems for automating human problem-solving capabilities. As examples of the latter, several rule-based expert systems for fault diagnosis and load scheduling have emerged. However, no single expert system capable of performing all the functions of power management has yet been developed. Presumably, the memory and computational demands exceeds the capabilities of a single, centralized expert system. In addition, it is highly unlikely that such a system could meet all the requirements for effective power system management, which includes a very fast response time for fault diagnosis and recovery. Such requirements are especially crucial in a hostile environment such as space, with limited redundancy capabilities in power system components. It is questionable, therefore, whether current-generation expert system technology is applicable for power system management for space. The objective of the ADEPTS project at JSC is to investigate new techniques for power management automation. These techniques involve integrating current expert system technology with that of parallel and distributed computing, as well as a distributed, object-oriented approach to software design. An overall design of the ADEPTS power management system was proposed as the result of the author's first summer fellowship [7]. The work performed during the second term, which is summarized in this report, is a further contribution to the ADEPTS development effort. The focus of the current effort is the integration of new procedures for automatically planning and scheduling loads with procedures for performing fault diagnosis and control, already implemented in ADEPTS. The objective is the concurrent execution of both sets of tasks on separate transputer processors, thus adding parallelism to the overall management process.

INTRODUCTION

The objective of the activity summarized in this report is the integration of planning and resource management into the ADEPTS power management system. Planning activity takes the form of scheduling, rescheduling and executing loads, and the automation of this activity should admit interaction with the human planner. Resource management consists of monitoring the supply of power to the overall system, configuration management, and load shedding.

The immediate purpose of this research activity was two-fold: first, to add more autonomy to ADEPTS, where by autonomy is meant the ability to perform control functions without the intervention of human control, and secondly, to further investigate the use of parallel processing for the performance of power management for spacecraft.

In addition to developing these ADEPTS enhancements, a more detailed overall design of a more robust ADEPTS system was constructed during this research activity. This work furthered the effort of last year's summer term (summarized in [7]), and is also summarized in this report.

ADEPTS PLATFORM

Introduction to ADEPTS

ADEPTS (Advanced Electrical Power Management Techniques for Space systems) is an example of a "reactive, real time system" [4, p. 997]. By reactive is meant a non-terminating program which maintains an on-going interaction with its environment. By real time is meant the imposition of timing constraints on the execution of the tasks of the system. [1, p. 1]. The environment here comprises both the power system itself, as well as the human operator (crew member or mission control personnel).

ADEPTS uses distributed memory and distributed processing for power system management. Independent, communicating processes cooperate to solve problems related to diagnosis, planning, and control of a power system. The memory is physically distributed across a network. ADEPTS instantiates a distributed process model involving synchronous message-passing. Processes interact using a rendezvous-like protocol to service requests for data. Currently, ADEPTS is implemented on a network of transputers using a Macintosh host.

ADEPTS seeks to incorporate recent innovations in object-oriented processing to the system design [1, 8]. The object-based paradigm is being used to establish a concurrent system development environment, whereby an object is viewed as an entity whose behavior is described in terms of operations it is subjected to and carries out on other objects [1]. This approach has also been used in the development of the ADEPTS

prototype in Ada [9].

Finally, the ADEPTS processing model incorporates techniques from distributed knowledge-based systems. It includes the idea of a small, loosely-coupled group of semi-autonomous managers (defined below) cooperating to continuously maintain the proper distribution of electrical power to a set of loads. The organization model loosely chosen as the paradigm for ADEPTS is the mission control model, wherein a committee of experts cooperate to maintain the proper flow of power through the spacecraft.

For clarity, the following terminology is used to describe the processing components of ADEPTS. ADEPTS is viewed as a network of managers. A manager is a processing unit which has been allocated a role (set of tasks, expertise). A role may consist of some expertise in power management (fault diagnosis, scheduling), data processing (e.g., providing an interface to the human operator) or communication and control. In order to perform a role, a manager must have direct (i.e. local, as opposed to remote) access to, and manipulate, a set of objects in memory; such objects include a database and knowledge base. A manager may also control a set of agents, which are smaller processing units dedicated to the performance of a single task among those allocated to the manager. From an architectural point of view, each manager in ADEPTS is assigned a single processor (specifically, a transputer) and some memory; it runs independently from the other managers. Agents local to the same manager run concurrently on the same processor and share memory. The result is a network of managers.

Currently, the overall power management task consists of the following roles:

* Load Management
* Fault Management
* Resource Management
* Low-level Monitor and Control Management
* User Interface

A more detailed description of each of these roles appears in [7]. The processing tasks involve a combination of data processing, knowledge management, numerical processing, and communication. The highlighted components in the above list indicate the scope of the investigation summarized in this report.

The current overall ADEPTS control architecture, including communication links, is displayed in Figure 1. Each of the managers runs on a transputer in parallel. As ADEPTS executes, math model data, representing voltage and current readings from a power system [6], is sent for evaluation and analysis to the power manager executive. Concurrently, loads are being planned and executed via the user interface and the

load manager. If the power system data indicates the presence of a fault, the fault manager is invoked to diagnose and implement corrective measures.

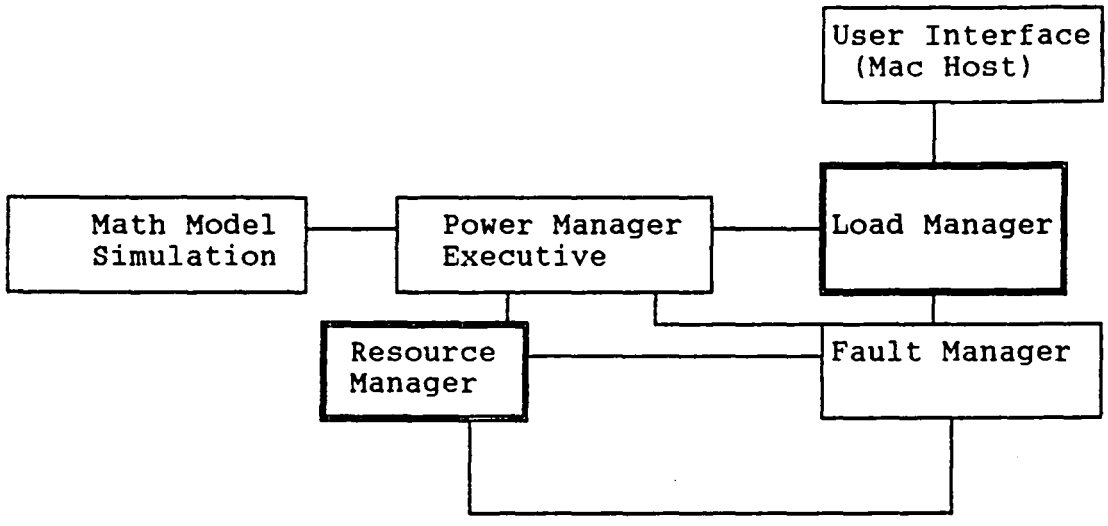

Figure 1: ADEPTS Configuration

The enhancement to ADEPTS which resulted from the project described in this report is the addition of the Load Manager and Resource Manager (represented in the figure by the double-stroke lines). The load manager was developed in order to automate the process of executing loads to a power system. The resource manager determines whether there is enough power to execute loads as planned. Finally, communication links were established between the load manager and the fault manager, the user interface, and the resource manager. Each of these components is described further in this report.

ADEPTS Software Engineering Approach

The life cycle of a real time system such as ADEPTS is complex. Figure 2 is a reproduction of (part of) a standard flow chart for an ideal real-time system life cycle [1,p.76]. ADEPTS is currently at the requirement specification phase. The result of the preliminary activity was the selection of the transputer as the architecture in which to develop ADEPTS, and the development of a small, experimental version of ADEPTS (summarized in [6]). The system requirement review was performed by initial NASA and contractor developers, and by the author of this report during last summer's research (summarized in [7]).

The software engineering philosophy of ADEPTS is that too much documentation, and not enough implementation is not a healthy approach. One reason for this is that the software and hardware technology being applied to power management has not

been extensively tested and verified. Another reason is that the developers do not have extensive programming experience on the transputer, and hence require extensive hands-on training. Thus, the developers prefer an approach which combines three activities during the current specification and review phases:

* Specification documentation
* Rapid prototyping of concepts
* Iterative development on the transputer

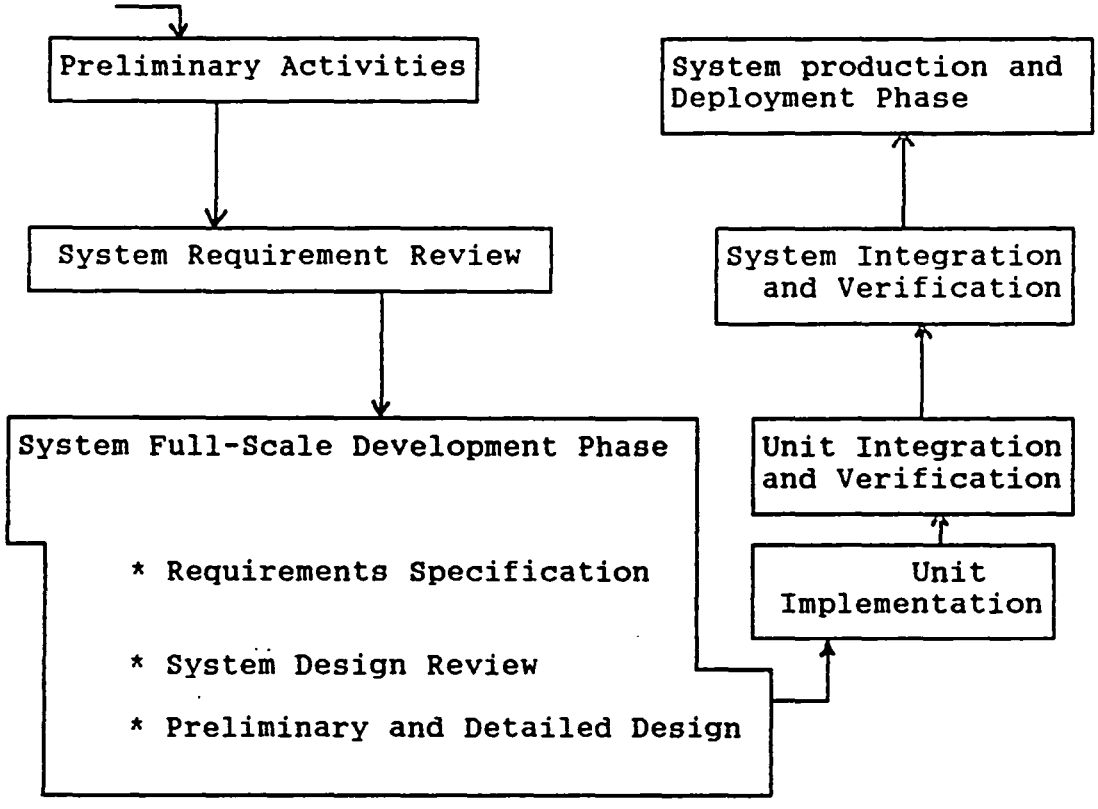

Figure 2. Real-Time System Life Cycle

The purpose of the first task is clear: to provide a detailed design and task description for ADEPTS, and to allocate subtasks to the ADEPTS developers. The purpose of the second task is to allow for a empirical verification of the overall design, and to identify any modifications or further enhancements to the design that might be required. A prototype environment for ADEPTS has been developed in Ada under Unix [9]. Finally, the purpose of the third task is to test the real time machine-dependent aspects of the design, especially the synchronization and communication mechanisms. Performing the third task also has the practical advantage of allowing the developers, many of whom may be unfamiliar with the transputer environment, with an opportunity to become more

familiar with its features.

LOAD MANAGER AND RESOURCE MANAGER DEVELOPMENT

In the following sections, we describe the steps leading to the development of the load management and resource management roles in ADEPTS. First, a set of requirement specifications for each manager is identified. Then, the data structures and algorithms used to implement these managers are described. It should be noted that the scope and complexity of each of these manager's roles prohibited an implementation of all the features specified in the requirements description in the ten weeks allocated for this activity. Thus, the discussion concludes with a list of future enhancements required for each manager.

Resource Manager Requirements Specification

The resource manager automates the power management tasks associated with system configuration, including

* Monitoring the supply of power to the system;

* Reconfiguration as a result of adding or deleting loads, or as a response to a faulty condition; and

* Load shedding as a result of too much demand for power.

The goal in developing this agent is to free the crew member and mission control personnel of routine power supply monitoring tasks, to have an effective intelligent control mechanism in the event of loss or decrease of power to the system, and to aid in the planning and execution of power system events.

To perform these tasks, a resource manager must interact with both the diagnostic and planning managers, as well as with the low-level power system monitors, who will supply it with sensor data from the main busses supplying power. The latter will allow the resource manager to determine whether a loss of power to the loads has occurred, and consequently whether shedding may be required.

The communication link with the planner (load manager) provides the resource manager with the task of filtering commands to the system for powering on loads. This filter is necessary to avoid requests for power to loads on a path which contains a fault. On the ADEPTS approach, there are two ways this scenario can be avoided. First, the load manager will be informed of faulty power system components. This will allow it to delay requests for power to such loads. The resource manager provides a second filtering mechanism; it will

prohibit situations in which a fault has occurred and identified by the fault manager, but has not been yet recorded by the load manager.

Consequently, the resource manager will act as a filter in case either there is a fault on the main bus, or there is a fault downstream of which the load manager has not been notified.

In addition to being a filter, the resource manager is also a configuration expert. It has the expertise to translate load requests into activation commands (i.e., throwing switches off or on). It also shares with the load manager information about the critical loads in the systems, those loads with redundant sources of power. If a fault on one of the paths to the critical load is detected, the resource manager will automatically reconfigure the system to the redundant path. This act may result in the lack of power to other loads, and hence the need for shedding loads, which will also be performed automatically by the resource manager. Finally, load shedding may need to be performed as a direct result of a reduction in the supply of power on the main bus; this event will be perceived by the resource manager, and the act of shedding performed.

To summarize, the resource manager has a three-fold role: as a monitor of the supply of power along the main bus, as a filter of load requests from the load manager, and as a configuration expert, which implies the ability to perform load shedding and redundancy switching for critical loads. In future enhancements to ADEPTS, each of these three sub-roles may be assigned agents.

Data Structures for ADEPTS Resource Manager

To perform its role, the resource manager maintains two lists as data: a fault list and an activation list. The fault list indicates which components of the system are faulty. The activation list indicates the status of each switch in the system (on or off). These lists are constantly being updated, as the result of its own actions (i.e. controlling the switches), as the result of diagnostic information from the fault manager, and from its own main bus sensor data indicating available power.

The expertise it has to allow it to translate load requests into switch commands resides in a configuration knowledge base. This consists of structural information about the configuration of the system, as well as load information (e.g. how much power a load needs, and whether or not it is a critical load, one with a redundant path).

ADEPTS Implementation of Resource Manager

The current ADEPTS resource manager does not perform load

shedding or redundancy reconfiguration. In pseudo-code, the current resource manager consists of the following algorithm:

```
loop
  {wait for
    fault_data; (* from fault manager *)
    load_request; (* from load manager *)
   case
    fault_data
         update_fault_list;
    load_request
       confirm_request; (* check for fault *)
       if confirmed then
        reconfigure_system; (*send switch commands*)
         update_activation_list;
  }
```

Confirm_request checks the fault list and configuration knowledge base for faulty conditions along a path to a requested load. Reconfigure_system constructs an activation command, which is a set of switch commands (on or off), which it sends to the executive, and ultimately to the math model simulation.

Future Enhancements to Resource Manager

The current ADEPTS Resource Manager implements the filtering task and part of the configuration task in fulfillment of this manager's requirements. To complete the implementation, the monitoring task must be developed, as well as the ability to perform redundancy configuration and load shedding. Finally, the load manager should receive from the resource manager an acknowledgement of its request, or, if the request cannot be met, a message indicating that the request cannot be serviced.

Load Manager Requirements Specification

The load manager is responsible for the planned control of power system events. A load manager should be flexible enough to operate either autonomously, or as an assistant to the human operator. "Offline", the load manager should manage the planning and scheduling of loads. "Online", it should contain the means to automatically cause the execution of a set of scheduled loads. In carrying out its assigned task, the load manager will free both crew member and mission control personnel from routine or time-consuming tasks related to planning and executing loads. In addition, it will provide an on-board database of load information, and procedures to assist humans in manually developing or changing load schedules.

A robust on-board load scheduler should combine both predictive and dynamic aspects. By predictive is meant the capability to plan a schedule for the future. By dynamic is meant the capability to react to unforseen events in the environment by rescheduling loads. An on-board system should also combine full autonomy with an interactive capability. It should assist the human in incrementally developing a schedule, and it should also be able to develop a full schedule from a list of loads and their constraints. The scheduling algorithm can combine traditional techniques with knowledge-based approaches. The most promising knowledge-based approach to predictive, autonomous scheduling is the constraint-based approach. This approach has been used in a number of applications, including payload scheduling for spacecraft [10]. Recently, a distributed constraint-based approach has been proposed [11].

Data Structures For Load Management

The ADEPTS load manager consists of a load table containing information about loads to be powered during a relatively short duration (one day) of time. Its permanent knowledge consists of information useful for either automatic or interactive planning and scheduling activity. Load priority, power and resource requirements, timing constraints, as well as whether the load is currently faulted, are examples of useful load information.

The load table could consist of a short term (one day) schedule of loads, or a long-term schedule, or both. The load event should consist of a load identifier, and the interval(s) during which it is to be powered, comprising a start time and end time. (The load table can be displayed to the user as a table of time intervals). Thus, for example, Payload L3 has been scheduled once during this time frame, during the interval starting at mission time 2 and extending to mission time 9.

The load table is implemented in ADEPTS as a linked list of load information. This information is of two kinds: status information and occurrence information. Status information is a record implemented in C as follows:

```
typedef struc
 { short activation status;
   short fault status;
   tslot timeslot; }
 loadrecord;
```

Activation status (OFF or ON) indicates whether a load currently has power. Fault status (FAULTED, HEALTHY) indicates whether there is a fault somewhere in the line leading to the load.

Occurrence information contains the start and end times of each scheduled occurrence of the load. This allows a load to have multiple, periodic occurrences during the same time frame. The C record for an occurrence has the following declaration:

```
typedef struc
 { ulong startTime ;
   ulong endtime ;
   tslot timeslot;(* pointer to next load
                     occurrence*)
 }
   tslotrecord;
```

These data structures should be enhanced in future work to include other information useful in the scheduling or rescheduling of loads. For example, load priority information, resource requirements (power, instruments, crew, etc.) could be declared as additional slots. Since the current load manager does not engage in any scheduling activities, it was not deemed necessary to include these fields in the definitions.

Load Manager Functional Specification

The load manager role can be classified into three subtasks:

* Monitoring/executing events from the load table
* Updating the load table
* Scheduling and rescheduling loads

These tasks can be viewed as independent; hence, as involving separate agents. On the other hand, each operates on the same data structure, the load map. Hence, for future ADEPTS enhancements, it should be possible to perform these tasks concurrently, but probably using one transputer.

The communication links required by load manager are to the User Interface and Resource Manager (Figure 1). The user interface link allows the user (crew member or mission control personnel) to update the load table as required. It also allows for the communication and display of load table and other load information to the human operator. The resource manager link allows for the coordination of load execution activity. As noted, the resource manager will not cause a load to be executed if a fault has been identified by the fault manager along a path leading to this load, unless a redundant path exists to this load, and there is enough redundant power to allow the load to be executed. To avoid the useless communication of load execution information between the load

manager and resource manager, the load manager can receive current fault information from the resource manager. This allows the load manager to update its load table so that loads along the path of a fault will not be executed until the fault has been corrected.

C Implementation of Load Manager Functions

The main procedure of the load manager currently consists of calls to initialize a time map and the local clock, and a continuous loop for monitoring and executing loads:

```
void main
{
      InitializeTimeBase ();
      InitializeLoadRecord ();

     do
      CheckSchedule ();
     while (DoCommand ());
}
```

InitializeTimeBase requests the real time from the main clock, and converts it into a "mission time" framework, where the 0th time point is the onset of the mission. This time measure corresponds to the metric used in the load map; i.e., each load map starts at 0 mission time. This simplified time framework suffices for the purposes of this "not fully real-time" version of ADEPTS. Future versions will demand more complex procedures for representing real time and mission time.

The do loop in the main procedure handles both the communication and time map monitoring. DoCommand is a procedure for input channel communication. The input will be in the form of schedule information, either from a file, or through user commands from the keyboard. CheckSchedule is the monitor and load execution procedure. It performs a traversal of the time map to search for loads that need to change their status (from on to off or visa versa). To match global time with mission time, a function for converting one to the other is employed. On the Mac, global time is depicted as number of seconds that have elapsed since midnight on January 1, 1904. This value is translated into mission time within Checkschedule.

If a load status needs to be changed, load information in the form of a load identifier and a command, is communicated to the resource manager.

Possible Enhancements to Load Manager

A number of enhancements is required for Load Manager to

meet design specifications. In sum, these include:

* Development of Predictive Scheduler
* Development of Enhanced Dynamic Scheduler
* A Load Map Display Mechanism
* A Finer Partitioning of Management Tasks
* A More Object-oriented Approach to Knowledge Representation
* Development of a More Interactive Approach to Scheduling (More User Friendly Control)

FUTURE OVERALL ADEPTS ENHANCEMENTS

In addition to the enhancements to the load and resource managers recommended above, additional development efforts have been identified as the result of this summer's activity. In this section, these are summarized.

Certain limitations of the Macintosh/transputer network have been identified and should be addressed in the future. First, major limitations of the transputer are due to its message passing protocol. In ADEPTS, the transputer hardware and software forces a synchronous exchange between processes. This means that the execution of an output command by a process i is delayed until the receiving process j is ready to execute a read operation, and visa versa. There are potentially serious problems as a result of imposing synchronism on processes. One of the most serious is the absence of fault-tolerance of transputer networks; if a receiving process fails, each of the processes which communicate with it will eventually fail as well. It should be noted that a form of asynchronism can be achieved through employing a timed-out mechanism in the communication. As ADEPTS develops, this mechanism should be considered more fully.

Additional limitations arise as a result of the interaction between the Macintosh host and the transputers. The host computer is required to simultaneously process user input and transputer data. The programmer is required to establish the protocol for servicing these requests. If the transputer is given full priority, the user may be indefinitely forced to wait on his or her request. On the other hand, if the user is given priority, the transputer is forced to wait until its request can be serviced, which means that its other processing tasks may be thwarted. The current solution employed in ADEPTS allows for fairness in the servicing of requests from the user and the transputer. As a result, however, the performance of the transputers may still be affected, especially if it is assumed that the user will use the system often.

Part of the problem is that the current Machintosh operating system (6.0) does not support multitasking

capabilities. As a result, the programmer is forced to ensure fairness and liveness properties of the system. Future versions of the Macintosh operating system (7.0) will support multi-tasking, and hence improve the real-time capabilities of the system, and free the programmer from the responsibilities of enforcing fairness and liveness properties. It behooves the developers of ADEPTS to address the problem of the host/transputer interaction in more detail.

The final set of enhancements to be recommended for future ADEPTS development activity address the furthering of the effort of establishing an overall design for a real time ADEPTS system.

In general, ADEPTS can be viewed as having four layers:

* The Interface layer
* The Communication and Control layer
* The Data and Numerical Processing layer
* The Expert System layer

Each of these layers requires a more detailed set of requirements than is currently available. The interface layer requires specifications for the user interface, as well as the lower-level interfaces to the system hardware. The communication and control layer demands a set of specifications for communication among managers, and among agents within a manager, as well as deciding on a more effective global control mechanism for the performance of management tasks. The data and numerical processing layer defines the data to be used and communicated among managers, as well as the algorithms for processing the data. Finally, the expert system layer addresses problems of knowledge base representation and management, handling uncertainty, and heuristic reasoning.

SUMMARY

The ADEPTS project seeks to integrate expert system technology with parallel and distributed computing for the purpose of developing an autonomous system for spacecraft power management. The primary focus of this summer's activity was on the means of increasing the autonomy of ADEPTS by developing techniques for the automatic scheduled execution of payloads, as well as a means of coordinating planning with limitations in power resources. Implementing these techniques also aided the developers in evaluating the transputer as a parallel computing hardware for ADEPTS. A secondary focus of this summer's work was to continue the effort of developing a set of design specifications for a more real time version of ADEPTS. The overall result of this effort has been a more precise and detailed description of the ADEPTS control mechanism and agent roles.

REFERENCES

1. Levi, S. and Agrawala, A. Real Time System Design. McGraw-Hill, 1990.

2. Hughes, J. Object-Oriented Databases. Prentice Hall, 1991.

3. Van Leeuwen, J. Handbook of Theoretical Computer Science, Vol. B: Formal Models and Semantics. The MIT Press/Elsevier, 1991.

4. Emerson, E.A., Temporal and Modal Logic. In [3], Chapter 16.

5. Lamport, L., and Lynch, N. Distributed Computing: Models and Methods. In [3], Chapter 18.

6. Sandman, T., Advanced Electrical Power Techniques for Space Systems, Phase I Report. Document JSC-24745, October 1990.

7. Morris, R., A Design for an Intelligent Monitor and Controller For Space Station Electrical Power Using Parallel and Distributed Problem Solving. Final Report, NASA/ASEE Summer Faculty Fellowship, Johnson Space Center, 1990.

8. Chin, R., and Chanson, S., Distributed Object-Based Programming Systems. ACM Computing Surveys, Vol. 23, No. 1, March 1991, pp. 91-124.

9. Morris, R., and Baggs, R. Distributed Intelligence Monitoring and Control. NASA-JSC Final Contractor Report, T-6956R, 1991.

10. Touchton, R.A. Common Module Dynamic Payload Scheduler Expert System. 21st Intersociety Energy Conversion Conference, Washington, DC, American Chemical Society, 1986.

11. Rzevski, G. Distributed Intelligent Knowledge Based Systems for Cell Scheduling and Control. ISATA 19th International Symposium on Automotive Technology and Automation, Monaco, 1988.